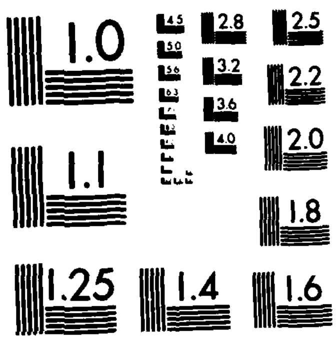
AD-A146 905 COMPUTER SCIENCE RESEARCH IN EUROPE(U) OFFICE OF NAVAL RESEARCH LONDON (ENGLAND) J F BLACKBURN 29 AUG 84 ONRL-R-10-84 1/1

UNCLASSIFIED F/G 9/2 NL

END

FILMED

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ONR LONDON REPORT

R-10-84

OFFICE OF NAVAL RESEARCH

BRANCH OFFICE
LONDON
ENGLAND

AD-A146 905

COMPUTER SCIENCE RESEARCH IN EUROPE

J.F. BLACKBURN

29 August 1984

UNITED STATES OF AMERICA

This document is issued primarily for the information of U.S. Government scientific personnel and contractors. It is not considered part of the scientific literature and should not be cited as such.

[illegible]

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: R-10-84
2. GOVT ACCESSION NO.:
3. RECIPIENT'S CATALOG NUMBER:
4. TITLE (and Subtitle): Computer Science Research in Europe
5. TYPE OF REPORT & PERIOD COVERED: Technical
6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): J.F. Blackburn
8. CONTRACT OR GRANT NUMBER(s):
9. PERFORMING ORGANIZATION NAME AND ADDRESS: US Office of Naval Research Branch Office London, Box 39, FPO NY 09510
10. PROGRAM ELEMENT PROJECT, TASK AREA & WORK UNIT NUMBERS:
11. CONTROLLING OFFICE NAME AND ADDRESS:
12. REPORT DATE: 29 August 1984
13. NUMBER OF PAGES: 12
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):
15. SECURITY CLASS. (of this report): UNCLASSIFIED
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:
16. DISTRIBUTION STATEMENT (of this Report): Approved for public release; distribution unlimited.
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report):
18. SUPPLEMENTARY NOTES:
19. KEY WORDS (Continue on reverse side if necessary and identify by block number):
Computers
Computer architecture
Computer programming
Robotics
20. ABSTRACT (Continue on reverse side if necessary and identify by block number):

Computer architecture, programming, and robotics represent the major new directions in European computer-science research. In computer architecture the emphasis is on large-scale computing. Functional and logic programming are receiving much attention. Sensing and control are featured in the robotics research.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

COMPUTER SCIENCE RESEARCH IN EUROPE

1 INTRODUCTION

This report surveys computer science research mainly in European universities and government laboratories. It is based on visits to a fairly large number of university computer science departments and government laboratories in the UK, France, West Germany, Italy, The Netherlands, Belgium, Sweden, and Spain over a period of 2 years. Also included is information gained from a number of conferences and workshops, including one on computer architecture and another on robotics sponsored by the Office of Naval Research and held in London.

The three topics highlighted in this report are computer architecture, robotics, and programming, since they represent the major new directions in computer science research in Europe. In computer architecture the emphasis is on large-scale computing. Sensing and control are featured in the robotics research. And functional and logic programming are receiving much attention in the programming field.

2 COMPUTER-SYSTEMS ARCHITECTURE

Interest in computer-system architecture in European universities and government laboratories has increased greatly within the last few years. Efforts to stimulate research and development in this subject have been made in the European Economic Community as an entity and in most of its individual members--especially Great Britain, France, and West Germany.

This increased activity is due in part to Japan's announcement of plans for a fifth-generation computing system and subsequent progress in that direction. However, it is principally due to more basic considerations: (1) the need for more computing power for solving problems in science and engineering, (2) interest in new applications such as image processing and expert systems, and (3) the steady advance of technology which makes it all possible.

The Need in Science and Engineering

A recent report of the panel on Large Scale Computing in Science and Engineering, following a study sponsored by the US Department of Defense and the National Science Foundation, lists areas of critical importance:

- Aircraft design. Aircraft are now designed in pieces because no computer can simulate the entire aircraft and the flow of air around it.
- Submarine design. The problem is similar to aircraft design.
- Geophysical exploration. Current computers can handle only simplified models of geological formations of existing or potential oil fields, for example.
- Atmospheric models. More detailed models of the atmosphere and the Earth's surface effects would dramatically improve short-range weather forecasts. However, more powerful computers are needed for this processing.
- Nuclear weapons. Drastic simplifications have to be made in the analysis and design of nuclear weapons in order to fit them to current computers.
- Electronic devices. More powerful computers than are now available are needed for three-dimensional design of circuit layouts for electronic chips. This would lead to better performance and lower cost for the chips.
- Command and control. More powerful computers would greatly improve the control of chemical refineries, power plants, and automated assembly lines. Also, adequate analysis and response to an attack on a US Navy ship or an Army unit would require a more powerful computer than presently available.

More powerful computers also could be used for controlling disease, handling nuclear power plant accidents, studying ocean circulation, developing magnetic fusion energy, and analyzing satellite images.

From these and many other examples in science, engineering, economics, and general management it is clear that the need exists for more powerful systems. The question, then, is how and why is it possible to achieve such systems.

The Technology for More Powerful Systems

Through improved techniques for making circuits smaller, faster, and cheaper, it is possible to build very fast general-purpose and special-purpose computers. There appears to be a limit to the attainable speed of a single processor, estimated to be somewhere between 10^9 and 10^{10} floating point operations per second. However, Gerola and Gomory (1984) suggest that this speed will evolve through constant improvement of circuitry, mostly through miniaturization, toward 10^{11} instructions per second.

Whatever may be the ultimate attainable speed of a single processor, much research and development effort in Japan, the US, and Europe is going into designing and building parallel processors to achieve more power in a system. Because of the relatively powerful and inexpensive microprocessor, universities have become very active in designing parallel computer systems using microprocessors as basic components.

One approach to a parallel system is a design that allows for a number of processors operating in parallel on different data but with the same operations being performed. A computing system with this structure is called a single instruction, multiple data (SIMD) machine. Such a parallel design fits some problems quite well, but certainly not all. The rigidity of the machine design in general allows only partial mapping of the problem onto the machine.

Another structure is the multiple instruction, multiple data (MIMD) system. One approach to this design is to have the processors that operate in parallel share the memory of the system. One difficulty with this is the increasing competition for memory access with the increase in the number of parallel processors in the system. Another approach to a MIMD system is to arrange for message passing via a crossbar from the parallel processors to multiple memory modules. However, the partitioning of the problem to be solved is a difficulty in both approaches.

Manchester University, UK

A somewhat less conventional approach to machine organization is that of data flow, the approach taken at the University of Manchester. It is also the approach taken by the Japanese in fifth-generation computing systems architecture.

A data-flow machine contains a number of independent processors; each receives a stream of data tagged with destination and control information. By matching tags, the processor sorts out the data due to be combined. On finding a match, the appropriate calculation is performed, and one or more new data packages (tokens) tagged with this destination are sent out. The processors are connected through a communications network.

The basic structure of the Manchester data-flow machines includes a processing unit, a token queue, a matching unit, a node store, and a host computer to provide peripheral control and storage. The units are connected in a pipelined ring around which the tokens flow. The tokens carry data, a label, and destination node addresses. A token produced by a node exits from the processing unit, where the execution of several node operations may be proceeding concurrently. Upon arrival at the token queue, the token is stored temporarily. The matching unit collects pairs of tokens with the same destination node address and label. If no partner is found, the token is written in the store to await a partner. A pair of matched tokens leaving the matching unit addresses the node store to obtain the destination node operation and the subsequent destinations of its outputs. The package is then passed to the processing unit for execution.

The processing elements in the processing unit have writable microprogram

storage so that instruction-set changes can be made readily. The processing unit has two pipeline stages. One handles label operations and the gathering of performance statistics; the other is a parallel array of processing elements. Each element of this array performs 24-bit integer or 32-bit floating point arithmetic. The microinstruction cycle time of each processor is 200 nanoseconds, and macroinstructions include between 5 and 50 microinstructions, with an average instruction time of about 6 microseconds. This requires 20 processors to match the input rate of one executable package every 300 nanoseconds.

Preliminary evaluation of the 20-processor system at Manchester has shown that a wide variety of quite small programs contain sufficient parallelism to exhibit impressive speed-up versus the number of active function units in a single-ring system. More work is needed to determine the behavior of large programs which cause matching store overflow. (See *ESN* 36-12:323 [1982].)

Imperial College, London

Another nonconventional multiprocessor system is the graph-reduction computing system being designed and built at Imperial College, London. The system will be used for processing symbolic expressions in optimizing high-level-language programs. It will be built from standard building blocks, the single-chip microprocessor called the transputer, made by INMOS Ltd. of Bristol. The plan is to have a large number of concurrent activities in the system. Whereas in a data-flow machine pieces of the program are activated by the availability of data, in a reduction machine the overall need for a solution causes functions to be applied in a more unpredictable manner.

The graph of an expression is represented by a collection of packets, each of which represents one node of the graph, the arcs extending down from that node, and necessary control information. A packet consists of three primary and three secondary fields. The primary fields contain the information required to represent a node, while the secondary fields contain the control information required for evaluation.

For a node to be reducible, it must be associated with a reducible function (one for which there are rewrite rules), and the arguments must match the left-hand side of some equation in the function definition.

The Imperial College system will be able to accommodate functional languages and so-called variable free languages.

A simulator (written in Pascal language) has been developed for the abstract machine. This simulator permits execution of programs compatible with the HOPE compiler, which has been written for the system. The developers estimate that a desktop-sized system with 20 processing agents will be capable of executing about 150,000 packets processed per second. This is viewed as the system building block for larger systems. A 20-processor system is expected to be operational by the summer of 1985. (See *ESN* 37-10/11:400 [1983].)

Linköping University

The need for picture processing, or more generally image processing, has arisen in applications ranging from satellite image analysis to robot vision. The University of Linköping, Sweden, has developed a special-purpose computer for image processing. The General Operator Processor is an MIMD system using parallelism and pipelining extensively. For the special applications for which it is designed, the system is expected to be 100,000 times faster than a nonparallel system using more conventional techniques. Image processing is needed for analysis of multispectral images, advanced computer graphics, and robot vision.

The Linköping system provides a unification of the processes involved, including techniques for edge and line detection, line description, texture analysis, and description of regions.

The image processor operates on gray scales, color, or multispectral

images of varied size. Conventional arithmetic and logical image operations are efficiently implemented.

Image information is structural; the information is provided partly by the structural relations between these data values. The General Operator Processor adapts to the data by doing the processing in a hierarchical structure.

A minicomputer version of the processor will be marketed by Context Vision, a Swedish company, and will be available by the end of 1984. (See *ESN* 37-12:444 [1983].)

Technical University of Berlin

Another novel architecture, called data-structure architecture, has been developed at the Technical University of Berlin. This concept has been implemented in a computer called STARLET. Systems using this architecture manipulate at the hardware level arbitrarily complex data structure objects as entities. Only the entire data-structure is referenced by name, and substructures or single data items of a data-structure object are accessed by the execution of access functions. Very fast, dedicated-access processors execute the access functions. The conceptual bottleneck, present in conventional systems, is avoided because the computer supports procedural or functional programming languages that allow the state of arbitrarily complex data-structure objects to be transformed by one complex operation, invoked by a single instruction of function application. The physical bottleneck does not exist since data items are moved to and from memory not by execution of move instructions, which would have to be fetched from memory, but by use of an address stream calculated by the access processor at high speed.

Thus the inherent parallelism of data-structure objects is exploited for parallel processing in the SIMD mode.

Specialized systems, including very fast image processors, have been derived from this system. An image processing system using this architecture is marketed by Robomation/Intelligence, Carlsbad, CA, under the designation IRI P256 Vision system. (See *ESN* 38-6:303 [1984].)

Other European Systems Architecture

Various parallel architectures have been investigated at Grenoble University, including pipeline, data flow, Lisp computers, and systolic arrays. Professor F. Anceau believes that the best non-Von Neumann architectures use single or cluster Von Neumann processor blocks. He believes the Von Neumann architecture gives the best adaptation to hardware possibilities.

At the University of Lille, France, research in parallel processing includes the development of communication tools and multiprocessors. Image processing is of great interest there, and algorithm-oriented, pixel-oriented, and object-oriented architectures have been investigated. A very fast machine called MAP has been developed for image processing with effective real-time capabilities. It contains 16 processors and 16 memory banks, with a switching network allowing access of any processor to any memory bank.

At Delft University a small MIMD system with one processing module having eight time-parallel subprocessors was completed in 1981. A much larger system is now being built with a maximum of 16 arithmetic/Boolean processing elements.

A pyramidal architecture for image processing has been designed at Pavia University, Italy. It is a multiprocessor pyramid architecture made of tapered layers of processors in which each layer constitutes a SIMD machine. Different layers may execute different instructions, making the total system a multi-SIMD processing system. (See ONR, London, Report C-2-84.)

3 PROGRAMMING

Programming research in Europe covers a very wide range of activities. Several examples will be briefly reviewed here. However, perhaps the single most interesting fact is the rapidly growing interest in logic programming. This will be reviewed as the last item

in the chapter, but will be given the most attention.

Distributed Systems

Newcastle University, UK

At the University of Newcastle the Newcastle Connection for Uniting Multiple UNIX operating systems has been developed and implemented (UNIX is an operating system developed at Bell Laboratories). The system of UNIX components joined by the Newcastle connection is called UNIX United. It is a distributed system functionally identical to a single UNIX unit. Functions such as file access, device access, and input-output control work across multiple machines. Issues of interprocessor communication and network protocols are hidden from the user. There is no need for special protocols for file transfer, virtual terminal remote job entry, and network mail. The system is in regular use at Newcastle with five Digital Equipment Corporation PDP-11s connected by a Cambridge Ring. (See *ESN* 36-10:238 [1982].)

INRIA

A project called SIRIUS was established in 1977 at the Institut National de Recherche en Informatique et en Automatique (INRIA), Roquencourt, France, to design programming systems to manage distributed databases. The first system, called SIRUS-DELTA, has the following features: (1) the user is not aware of the data distribution, (2) various types of data distribution can be processed, (3) the system is reliable, and (4) heterogenicity in hardware and software is allowed. The system is operational. Following SIRIUS-DELTA, INRIA designed a system to implement a multi-database approach. Any database is defined according to some data model. For example the relational database model considers that a database is a set of relations. A multi-database is a set of databases or of multi-databases having the following properties: (1) a language to express manipulation of data that are not within the same database, (2) a language to define data within the multi-database and its structure, and (3) the dependencies between databases and multi-databases.

Having completed a multi-database system for distributed data management, INRIA is now working on a real-time distributed computing system. Such a system controls (via activators) and monitors (via sensors) a physical process that is governed by a given set of internal laws. (See *ESN* 38-3:127 [1984].)

Stuttgart University, West Germany

The research group at Stuttgart University has introduced formal, abstract models for describing modern dialogue concepts offered by high-level user interfaces. The dialogue concepts are menus, forms, and windows (logical screens). Using these abstract models they achieve a formal definition of system states and state transformations which represent the basics for describing dialogue interfaces. In this way a formal definition of the semantics of user activities can be given. Work has been completed on communications requirements of distributed database systems, protocols for checking the availability of remote sites, and replicated data and stable storage in distributed database systems. (See *ESN* 38-4:185 [1984].)

Free University of Amsterdam

An operating system for distributed database application has been devised at the Free University of Amsterdam. The system, called Amoeba, uses the capabilities approach for user authentication. A conceptual matrix is presumed in which each row corresponds to a user and each column corresponds to an object known to the system. The intersection of a row and a column tells what access, if any, the specified user has to the specified object. The system has a list, indexed by user, telling which objects the user may access and how. In Amoeba the basic elements are processes, ports, and packets. Processes communicate with each other by exchanging packets through their ports.

The interface between the operating system and the database system is defined by a small set of primitive instructions, of which PUT and GET are the most important. Higher level primitives like PUTREQUEST and GETREPLY are implemented by user-space library routines. (See *ESN* 38-7:368 [1984].)

IBM-Heidelberg and Karlsruhe University, West Germany

A joint project between Karlsruhe University and the IBM-Heidelberg Scientific Center is designing and implementing a prototype of a distributed academic and scientific computing system which supports the hardware and software required by students, faculty, and research staff in a university or research organization. The hardware in the academic environment is a heterogeneous collection of computers which are nodes in a network. Each node is controlled by an operating system which has to be extended to provide functions for communication between the nodes. Any node should be able to serve a selected group of users independently of where the service is implemented. It should be possible to add or delete nodes without updating central catalogues. Data security and reliability will be supported and monopolization of resources by end users will be prevented. (See *ESN* 38-4:180 [1984].)

Special Applications

University of Rome

The Dipartimento di Informatica e Sistemistica has developed a program for generating a model of industrial robots in symbolic form. The code, called DYMIR, derives the equations of motion of robots, considered as a chain of rigid bodies connected by suitable single-degree-of-freedom joints. The code can run on a variety of large computers, is transparent to the user, permits the calculation of dynamical models with up to 12 degrees of freedom, and takes into account elasticity and transmission losses at joints. (See *ESN* 38-1:12 [1984].)

Concentration Heat and Momentum, Limited, London

This firm has developed a computer package called GRAFFIC for the interactive graphical representation of fluid-flow phenomena. It is a three-dimensional package which provides interpretation and display facilities for numerical predictions of phenomena involving fluid-flow and heat/mass transfer. It is designed for use as a pre- and post-processor for solution methods of the finite domain or the finite difference type. GRAFFIC operates interactively, using a storage-type graphics terminal with associated hard-copy units as required. Communication with the user is by a series of meaningful prompts designed to make the operation self-explanatory. Facilities provided include representation of the flow geometry and solution mesh, contour maps and surfaces of scalar fields, and vector fields represented either by vector maps or selected planes, or by vector lines or streamlines. (See *ESN* 37-7:253 [1983].)

Polytechnic of Wales

A computer program for analyzing thin rotational shells subjected to static and dynamic loads has been developed at the Polytechnic of Wales. The program uses the finite element method for vibration analysis of elastic rotational shells. An efficient eigenvalue-economizer has been developed to reduce iteration time. The routine, which runs on microprocessors, can select the masters analytically at a given cut of frequency. It has been tested on a long cylindrical shell analyzed as a beam, a short cylinder fixed around both edges, vibration of cones in flexure, bending vibrations of thin disks, and natural frequencies of the cooling tower. (See *ESN* 37-7:253 [1983].)

University of Exeter, UK

A computer program has been developed based on a finite element model using equilibrium elements within which stress fields are defined in equilibrium

with edge stress resultants. It is suitable for plane stress or strain problems with linear elasticity. The edge stress resultants are classified as basic (normal force, shear force, and moment) and higher order. The higher order stress resultants supplement the basic class and involve edge stresses that are in self-equilibrium. The stress resultants can be treated as element unknowns, and the model can be analyzed using the flexibility method for finite elements; statical indeterminancy and procedures for solution of corresponding force paths are defined. Force transmissions between elements are through corner nodes.

The program incorporates triangular elements having six or nine degrees of freedom of edge stress resultants and the selection of suitable force paths from an interior graph embedded in the model. Algorithms for path selection are based on topological properties of the graph.

Many further examples of applications programs could be given. However, the above limited sample is representative. (See *ESN* 37-7:253 [1983].)

Artificial Intelligence and Expert Systems

Linköping University, Sweden

The Artificial Intelligence Laboratory at Linköping concentrates on representation and manipulation of knowledge, problem solving, and natural-language communication. Emphasis is now on mechanical engineering, partly because artificial intelligence's (AI's) emphasis today is on expert systems. The objective is to analyze a human expert's behavior until a computable theory of his expertise emerges. It is difficult to find out what information is involved at each point in the expert's behavior.

The work at Linköping concerns numerically controlled lathes. This example offers problem solving with highly interactive subgoals, explicit reasoning about problem-solving strategies, resource-sensitive planning, data dependencies, geometric reasoning, and highly pragmatic communication. Four specific projects are under way. One deals with abstraction and representation in automatic planning. The main problem is how to manage the huge amounts of knowledge necessary to solve realistic planning problems.

An exploratory study with the main goal of providing participants with experience about work with expert systems is also under way. Specifically, the problem of choosing the appropriate clamping method for an arbitrary workpiece to be turned on the lathe is being examined.

The third project is concerned with dependency nets and nonmonotonic logic. The laboratory is concentrating on the design of dependency nets and on their use for knowledge representation and influence. The update algorithm is the central algorithm of a dependency net. A completely new algorithm has been developed for the nonmonotonic update problem.

The goal of the fourth project is to develop models of human verbal communication. The laboratory's theoretical research will be accompanied by the implementation of an experimental natural-language interface. With such an interface, a computer system should be able to follow a user's spoken instructions. (See *ESN* 38-8:303 [1984].)

Kaiserslautern University, West Germany

Professor Paul Raulifs at Kaiserslautern says there are three requirements to consider when designing an expert system:

1. Expert systems are intended to manipulate vast quantities of poorly structured knowledge and skills. Therefore, expert systems require a representation of knowledge and skills which supports: (a) rapid detection of expertise from situations in which the expertise is immediately applicable to achieving goals, and (b) the acquisition of large amounts of new knowledge.

2. Expert systems should incorporate all domain-specific reasoning mechanisms and problem-solving skills

belonging to the respective area of expertise.

3. Expert systems should be able to comprehend clients' queries and to construct, explain, and apply solutions.

The work on expert systems at Kaiserslautern is in automatic programming; medical and technical applications; and planning, configuration, and construction of computer systems.

In the automatic programming effort a LISP language program is generated from abstract specifications. The system, called automatic programming expert (APE), constructs executable and efficient programs from: (1) algebraic specifications of abstract data types, and (2) abstract algorithms given as conditional term-rewrite rule systems with terms built up from operation symbols of the abstract data types involved.

An expert system for diagnosis in internal medicine starts with the information a physician gathers on a first visit. The system guides the physician through subsequent diagnosis to test for verification of the initial hypotheses. The system includes risk and cost analyses.

Another system is being developed for Daimler-Benz to be used for fault detection in engine tests. A system for planning, configuration, and construction will be used in civil engineering for conceptual design. (See *ESN* 38-6:305 [1984].)

Queen Mary College, London

The following expert systems have been developed at Queen Mary College:

1. A formal analysis of design-rule checkers for integrated circuit masks.
2. An information-retrieval system using fuzzy logic.
3. An expert system for maintenance and manufacturing.
4. An expert system for medical applications.
5. Logic and knowledge representation using fuzzy logic.
6. A self-organizing controller for a robot arm.
7. Self-organizing, rule-based controllers for cement mills.

See *ESN* 38-7:364 (1984) for more about the work at Queen Mary College.

Logic Programming

The first computer programming language was the binary language of the machine, called machine code. The next step was to develop symbolic rather than numerical machine code, which was called assembler language. After that came a series of higher level languages, including Fortran, Cobol, Basic, PL1, APL, Algol, and more recently Pascal and Ada. In all of these programmer must describe exactly ho result is to be computed. Such prog s are largely made up of command which specify actions to be performe . are called imperative programming l..guages.

Logic-programming languages are more descriptive than are imperative-programming languages but may contain imperative statements. Such programs are primarily descriptive definitions of a set of relations or functions to be computed. The execution of such a program involves finding an output corresponding to a given input. The first logic-programming language, Prolog, was developed in Europe. It was first done by A. Colmerauer (France) and improved by R. Kowalski (UK). Considerable research in logic programming continues in Europe. A few examples are discussed below.

Uppsala University, Sweden

A Prolog program has been written to generate a speed-optimal unifier program to unify data structures, where a Prolog interpreter spends roughly half of its time. The generating Prolog program is a declarative partial description of the unifier. The method has been applied to an experimental interpreter.

Another effort is directed toward improving the efficiency of logic programs without sacrificing their logical

purity. Predicates for creating and manipulating arrays have been introduced into LM-Prolog, a Prolog dialect running on Lisp machines. These predicates are implemented in terms of physical arrays and vertical arrays in a manner that is transparent to the user. For some users of these predicates, a compiler can produce code performing array references and updates that is as good as that produced by compilers for traditional programming languages.

A third effort uses symmetry in the derivation of logic programs. The formal development of a Horn-clause logic program implies a derivation of program clauses from a set of definitions, data structures, and computable functions given in full predicate logic. Each program clause in a logic program is derived separately from the definitions. Although the derivations differ mainly in structure for the different clauses, in some cases two program clauses in a program are similar and can be transformed into each other. A technique has been developed at Uppsala to avoid constructing both the derivations. Also, a method of detecting whether there is an analogous program clause for a derived program clause has been determined.

Heriot-Watt University, Edinburgh

An architecture to support the parallel execution of logic language has been developed at Heriot-Watt. The language Parlog (a parallel-logic programming language) has been used for the implementation. The language's "don't care" nondeterminism, which allows both AND- and OR-parallelism, returns only one solution. The control structure uses processes that build an and/or tree tailored for guarded clauses. A unification algorithm is introduced which solves the problem of multiple occurrences of a variable.

Imperial College, London

The Parlog language, which features both OR- and AND-parallelism, was designed to simulate a system by a network of parallel processes communicating by messages. Real time is replaced by a central simulated clock. The language Parlog, based on Horn clauses, differs from Prolog in two significant ways: its use of "don't care" nondeterminism and the use of modes. These added features make possible the concurrent evaluation of conjoined relation calls--i.e., AND-parallelism with stream communication between the calls. Each relation call is evaluated as a process. Shared variables act as one-way communications channels along which messages are sent by incremental binding to lists.

Louvain la Neuve, Belgium

A technique has been developed at Louvain to control search in logic programming by the addition of restrictive predicates to rules so as to cut off all blind alleys without losing possible results. Criteria are given to ensure that additional premises in clauses allow results to be determined without the use of trial-and-error procedures. These criteria require neither the introduction of special well-orderings nor the induction of limits of predicates. They take into account structural properties of bounded-length compositions of the original clauses.

Research Institute of Applied Computer Science, Budapest

This institute has developed a logic-programming language called Lobo, which is nearer to traditional languages than is Prolog. Both languages are capable of use for implementing the same algorithms. Lobo does not use pattern matching, is easily compiled, and can use the traditional features of programming.

Institute for Co-ordination of Computer Techniques, Budapest

A generalized data-flow model and its applications for constructing a highly parallel Prolog interpreter has been developed at Budapest. The parallel Prolog interpreter makes use of the advantages of OR- and AND-parallelism. Transformation of the AND/OR tree into a data-flow graph has been done. The

operator types needed for parallel evaluation of Prolog programs have been determined.

Telecommunication Laboratory and Study Center, Turin, Italy

At Turin a system has been implemented in Prolog to verify dynamic properties of concurrent processes.

Descriptions of concurrent processes with asynchronous communication can be checked against dynamic behavior specifications expressed by temporal logic formulas, under the hypothesis that the whole concurrent system can be modeled by a nondeterministic finite automation.

The basic components of the verifier are the model checkers for the chosen temporal logics, the symbolic simplifier, and the dynamic semantics of the description language.

The most active research in logic programming in Europe is found in the UK, France, Sweden, and Hungary. Germany is more heavily committed to Lisp, particularly at Kaiserslautern University. A great deal of work is under way in the design and implementation of expert systems, much of which began before Prolog came into being. (See ONR, London, Report C-4-84.)

4 ROBOTICS

The key research fields in robotics are sensing (visual and tactile) and control. The key development of the microprocessor and its use in the control of robots has greatly expanded the range of use for robots. Most universities, government laboratories, and companies concerned with robotic research are working in sensing and control. Robotic control brings together theories from computer graphics, kinematics, dynamics, control, and programming. The present-day robot evolved from the teleoperator and the numerically controlled machine tool.

The two leading manufacturers of robots in Europe are ASEA in Sweden and Kuka in West Germany. Two major automobile manufacturers in Europe also make large numbers of robots, mainly for their own use in automobile manufacturing. These are Volkswagon in Germany and Renault in France. In 1983 ASEA sold more than 1000 robots at prices ranging from $30,000 to $90,000, and Kuka sold about half that number. Volkswagen has approximately 1000 robots of its own manufacture in use in its automobile plants.

Research and development in robotics is important in France, Germany, the UK, Italy, and Sweden. Several examples of typical research projects are given below.

INRIA

The major emphasis at INRIA in the robotic field is on visual sensing. Image analysis and understanding have been largely concerned with two-dimensional problems. However, at INRIA the need for three-dimensional scene analysis has been recognized and is a major research effort there. Among the most promising sensors are the geometrical sensors which provide, in a reference frame, the Cartesian coordinates of points lying on the surface of an object. The sensors also give some intrinsic information about the shape of the object.

Most sensors use a structural light such as a laser beam. However, other ways of obtaining geometrical information about shapes include tactile, optical, and acoustical devices. By combining a large number of points on the object with other information--such as the normals to the surface at the points--the shape of the object can be determined.

INRIA has developed a polyhedral representation of objects. An object is represented by a set of three coordinates of points lying on its surface, with a spatial data structure linking them, precisely defined as a graph whose vertices are the measured points and whose edges join the points. A minimal structure is a polyhedron with the measured points as vertices. Polyhedra can be used to approximate any shape.

The shape of the object can be determined based on triangulation and an algorithm developed by J.D. Boissonnat of INRIA to represent the volume and the surface of the object. Many applications can be based on this knowledge, such as determining grasping areas and recognizing the object. (See *ESN* 38-8:428 [1984].)

<u>Laboratoire D'Automatique et D'Analyse des Systemes, Toulouse, France</u>

Autonomous Mobile Robot

A nonspecialized mobile robot called Hilare is equipped with multisensor systems and uses a multilevel computer and decision system. The machine has microprocessors for locomotion control, ultrasonics, camera control, image preprocessing, and communication. Higher level processing of information is by a larger local computer; back-up computers are at Paris and Montpelier. Hilare is intended to serve as a experimental support vehicle.

The perception system for the three-dimensional universe of the robot combines information received from a two-dimensional video and from a laser telemeter giving the depth. Two coordinates of a point can be derived from the tilt of the mirrors used to deflect a laser beam onto the target. The third coordinate is determined by the travel time of the laser beam.

The system must be able to find the position and orientation of obstacles, which are always treated as polyhedrons. The system must also be able to recognize objects in a room, a capability requiring complex pattern recognition.

The video signal from the camera is coded over several levels of grey. From the coded image, the contours of objects and adjacent objects are determined. The information from the camera is transmitted to a higher level of the computer and decision system and combined with the laser range information. The telemeter is coupled to the camera so that their optical axes are in agreement. The entire apparatus is mounted on a turntable on the robot. Commands to the motors used to deflect the laser beam are based on data from the microprocessor associated with image processing (INTEL 8030) or from the the higher level decision center. The number of measurements by the telemeter are reduced through using only information in the zones when a change of direction of a contour is detected; thus, corners of an obstacle or object are located.

All obstacles are assumed to be polyhedral and are represented in two dimensions by polygons or parts of polygons. The main objective in path planning is to create a structure composed of polygonal cells and to construct a graph connecting those cells to transform the geometric model into a topological one. The four steps are perception, space structuring, path search, and path execution.

A system to navigate a mobile robot has been written in APL, a programming language, and implemented on an IBM 3033 linked to a 16-bit minicomputer (MITRA 15). The microcomputer is in turn linked to the on-board, five-microcomputer structure (INTEL 8085) controlling the robot's motors and sensors. The system has been tested for many obstacle configurations. (See *ESN* 38-8:428 [1984].)

The development of DYMIR, a code to derive the equations of motion of robots--considered as a chain of rigid bodies connected by suitable single-degree-of-freedom joints--was mentioned earlier. Much other research in sensing and control is being carried out in Europe and has been extensively covered in articles published in *European Scientific Notes* in 1982-84.

<u>NATO</u>

The North Atlantic Treaty Organization has recently formed a panel which, over a 6-year period, will monitor trends and encourage international collaborations in sensory systems for robotic control. This will provide a framework for the large range of research projects in many disciplines which are expected to contribute collectively to the widespread application of

sensor-guided robots. This panel held its first meeting in Brussels on 26 and 27 March 1984 and its second meeting in London from 12 through 14 September 1984.

5 CONCLUSIONS

Basic research in computer science in Europe, including robotics, is on a level approximating that of the US and Japan. However, in general Europe is far behind the US and Japan in applying the research efforts in the production of products. In computer architecture the data-flow concept has been pioneered in the US (Massachusetts Institute of Technology) and the UK (Manchester University) but has been adopted by Japan as the basic architecture of its fifth-generation computing system. In programming, the logic programming language Prolog was originally developed in France and subsequently improved in England. It has been adopted as the language of Japan's fifth-generation computer project. Researchers in the robotics field in Europe have on many occasions been invited to Japan to lecture and to make recommendations.

One can give examples of highly successful application of research to products. However, they are far fewer than those that could be cited in the US or in Japan. Ferranti in the UK has been highly successful in manufacturing and selling semicustom microchips for complex applications in computers, cameras, televisions, and telecommunications. ASEA in Sweden has become the largest European supplier of robots and has successfully marketed its robots in the US and Japan. ASEA has about 25 percent of the European market in robots. INMOS, Ltd. in the UK has developed a complete computer on a chip, the transputer, which shows great promise in the marketplace. Imperial College, London, will use the transputer as its basic building block for the graph reduction computer scheduled for completion by the summer of 1985. Sinclair Research in the UK has been highly successful in marketing microcomputers and personal computers.

Indications are that government support for research in computer science will continue to be strong. The European Economic Community recently agreed to fund the European Strategic Program for Research and Development in Information Technologies (ESPRIT) in the amount of $1.3 billion over 5 years. The UK has its Alvey Program funded at £350 million over 5 years. Four main technologies will make up the Alvey Program: very large scale integration, software engineering, expert systems and intelligent knowledge-based systems, and man-machine interfaces.

France and West Germany have very similar programs at approximately the same level of funding. These have been covered in detail in *ESN* 37-12 (1983) and 38-1, 38-2, 38-3 (1984).

Reference

Gerola, H., and R.E. Gomory, "Computers in Science and Technology: Early Indications," *Science*, Vol 225, No. 4657 (6 July 1984).

END

FILMED

11-84

DTIC